VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Online verze karetní hry BANG!

Online Version of Card Game BANG!

2011 Jan Schovánek

Souhlasím se zveřejněním této bakalářské práce dle požadavků čl. 26, odst. 9 *Studijního a zkušebního řádu pro studium v bakalářských programech VŠB-TU Ostrava*.

V Ostravě 6. května 2011 .....

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 6. května 2011 .....

Rád bych na tomto místě poděkoval všem, kteří mi s prací pomohli. Předně bych chtěl poděkovat panu Ing. Lumíru Návratovi za jeho pomoc při vzniku této práce. Dále pak svým nejbližším za jejich podporu.

Abstrakt

Cílem mé bakalářské práce je vytvořit webovou aplikaci, která odpovídá karetní hře BANG!. Hra je určena pro minimálně 4 hráče a lze ji hrát po síti. Aplikace sestává z několika částí. Web pro založení hry a přihlášení k ní, aplet, ve kterém hra probíhá, server, který zpracovává veškerou logiku, a databáze pro uložení potřebných informací. Práce obsahuje seznámení s hrou, specifikaci požadavků a analýzu. Popisuje implementaci jednotlivých částí aplikace a průběh jejího testování. K vytvoření aplikace byla použita platforma Java EE 6 a nová technologie JavaFX, které bude věnována větší pozornost. Klientská část aplikace využívá princip tenkého klienta.

Klíčová slova: BANG!, Java EE 6, JavaFX

Abstract

The aim of my bachelor thesis is to create a Web application that corresponds to the card game Bang!. This game is for minimally 4 and maximally 7 players and it can be played via network. The application consists of several parts. Web sites for creating a game and logging to it, applet where is placed the game, server that handles all the logic and database for store the necessary information. The thesis includes introduction with the game, specification of requirements and analysis. It describes implementation of parts of application and the course of testing. For creating this application was used Java EE 6 platform and a new technology called JavaFX which will be given more attention. The client aplication uses principle of the thin client.

Keywords: BANG!, Java EE 6, JavaFX

Seznam použitých zkratek a symbolů

CSS	–	Cascading Style Sheets
ERD	–	Entity Relationship Diagram
F3	–	Form Follows Function
GPL	–	General Public License
GUI	–	Graphical User Interface
HREF	–	Hypertext REFerence
XHTML	–	Extensible HyperText Markup Language
Java EE 6	–	Java Platform, Enterprise Edition 6
JRE	–	Java Runtime Environment
JSF	–	Java Server Faces
JSON	–	JavaScript Object Notation
RIA	–	Rich Internet Applications
SDK	–	Software Development Kit
TLD	–	Tag Library Description
UC	–	Use Case
W3C	–	World Wide Web Consortium
XML	–	Extensible Markup Language

Obsah

Seznam tabulek

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

Bang! je karetní hra italského autora Emiliana Sciarra z prostředí divokého západu. Děj hry je zasazen do města plného přestřelek, ve kterém se šerif snaží město uchránit před bandity, kteří jej chtějí ovládnout.

Hra je určena pro 4 až 7 hráčů a každý hráč hraje za roli, kterou si vylosuje. Může si vylosovat roli šerifa, jeho pomocníka, bandity nebo odpadlíka. Cíl hry je pro každou roli jiný. Bandité vyhrají, pokud zastřelí šerifa. Odpadlík vyhraje v případě, že jsou mrtví všichni bandité a zastřelí šerifa. Šerif vyhraje, pokud jsou mrtví všichni bandité i odpadlík, a pomocník vyhraje tehdy, když vyhraje šerif. Všichni hráči znají svou roli, nevědí však, za jakou roli hrají ostatní s výjimkou šerifa. Jeho role je známa všem. A tak se zpočátku střílí naslepo, až se postupem času zjistí kdo má jakou roli, a nastává boj s časem, kdo koho zastřelí prvního.

Bang! je hra, ve které se hráči neobejdou bez vzájemné komunikace mezi sebou a také jejich pozornosti. Lidé jsou u ní schopni trávit spoustu času a odměnou za to jim je dobrá zábava. Díky tomu si našla své příznivce po celém světě.

Implementace her jako webových aplikací přináší výhody, mezi které patří především její dostupnost odkudkliv, kde je připojení na internet, a také odpadá nutnost instalace aplikace na uživatelův počítač. Díky těmto výhodám se webové aplikace v poslední době stávají velice rozšířené a těší se velké oblibě.

Postupně je v práci rozepsána specifikace požadavků celého systému, analýza, návrh a implementace, kde bude bližší seznámení s novou technologií JavaFX, a nakonec testování. Část testování je věnována také nasazení aplikace.

2 Specifikace požadavků

Tato kapitola popisuje bližší specifikaci požadavků, které jsou kladeny na systém. Jedná se o požadavky funkční, které musí systém splňovat z hlediska své funkčnosti, a také požadavky nefunkční, které jsou důležité pro nasazení systému na server.

2.1 Funkční požadavky

Funkční požadavky systému jsem pro přehlednost rozdělil na dvě části. První část pojednává o funkčnosti menu aplikace a druhá o funkčnosti hry samotné.

Systém bude sloužit jeho uživateli. Tím může být kdokoli, kdo navštíví web a přihlásí se pomocí přezdívky, kterou si sám zvolí. Přihlášený uživatel bude moci vytvořit novou hru, které přiřadí název a zapne nebo ponechá vypnutou kontrolu pravidel. Pokud bude chtít, může hru smazat. Toto bude umožněno na stránce s výpisem založených her. Na této stránce si také může uživatel vybrat, kterou hru chce hrát. Až nalezne požadovanou hru, přihlásí se do ní a bude čekat na přihlášení ostatních hráčů. Během čekání bude systém ukazovat stav přihlášených hráčů, a také bude umožňovat odhlášení ze hry. Po přihlášení všech hráčů se bude moci přesunout na stránku s aplikací, ve které bude probíhat hra samotná. Ze systému se bude také možno odhlásit.

Funkční požadavky na hru se týkají toho, co by měl systém umět během hraní hry. Hra se musí řídit svými pravidly. Jejich úplný popis je uveden v příloze a zde jsou pouze ta základní.

- Hru hraje 4 až 7 hráčů.
- Každý hráč má jednu postavu a jednu roli (šerif, bandita, pomocník nebo odpadlík). Postavy jsou známé všem, role, s výjimkou šerifa, nikoli.
- Na začátku hry má každý hráč tolik karet, kolik má jeho postava nábojů.
- Počet životů na začátku hry je roven počtu nábojů postavy. Šerif má ale jeden život navíc.
- Na začátku tahu si každý lízne dvě karty z balíčku.
- Vlastnost uvedená na kartě postavy je platná po celou hru.
- Šerif a jeho pomocník vyhrávají, pokud jsou zabiti bandité a odpadlík.
- Bandité vyhrávají pokud je mrtev šerif.
- Odpadlík vyhrává, pokud jsou zabiti bandité a po nich i šerif.

Jak již bylo zmíněno výše, v menu aplikace (při zakládání nové hry) je možné zapnout kontrolu pravidel. Touto kontrolou se myslí zbývající pravidla týkající se hraní hry. V případě zapnutí kontroly bude aplikace sama hlídat, zda pravidla hráči dodržují.

Hra bude také reagovat na stav přihlášených hráčů. Pokud dojde k přerušení spojení mezi serverem a některým hráčem, server tuto událost dá na vědomí zbývajícím hráčům a ukončí hru. Taktéž při výpadku spojení klienta se serverem klient upozorní hráče.

Vzhledem k tomu, že se jedná o hru, ve které je důležitá komunikace hráčů mezi sebou, bude v aplikaci dostupný jednoduchý chat. Dále je požadován přehled tahů hráčů a návod jak hrát.

2.2 Nefunkční požadavky

Bude se jednat o online webovou aplikaci, napsanou pomocí platformy Java EE 6 s využitím tenkého a tlustého klienta, konkrétně platforma JSF 2.0. K uložení potřebných dat bude využívána databáze MySQL. Hra samotná bude probíhat v okně webového prohlížeče a bude implementována v technologii JavaFX. Také je požadováno intuitivní ovládání a přehledné zpracování.

3 Analýza

V této kapitole se nachází analýza celkové aplikace. Skládá se ze dvou částí. Nejprve je popsána analýza dynamická, která popisuje chování systému při reagování na události, a poté analýza datová, která popisuje uložení dat.

3.1 Dynamická analýza

Aplikace reaguje na události vyvolané funkcemi, kterými disponuje. Tyto funkce byly pro přehlednější popis v minulé kapitole rozděleny na funkce systému (webové rozhraní) a funkce hry a tohoto rozdělení se drží i tato kapitola.

3.1.1 Funkce systému

Funkce systému byly popsány v předchozí kapitole. Tento popis byl úplný a zde bude pouze doplněn o diagram případu užití (Use case diagram) a diagram aktivit (Activity diagram).

Diagram případu užití se nachází na obrázku 1 a zachycuje možné funkce systému.

Diagram aktivit mezi menu, apletem a serverem se nachází na obrázku 3.

3.1.2 Funkce hry

Následuje popis funkcí hry, které umožňují hraní hry. Jsou to následující funkce:

- Braní karty - z lízacího balíčku, z odhazovacího balíčku, od jiného hráče,
- odhození karty - do odhazovacího balíčku, do lízacího balíčku, jinému hráči,
- akci na jednoho nebo všechny hráče,
- přidání a ubrání životů,
- zobrazení historie tahů,
- jednoduchý chat - psaní a přijímání příspěvků.

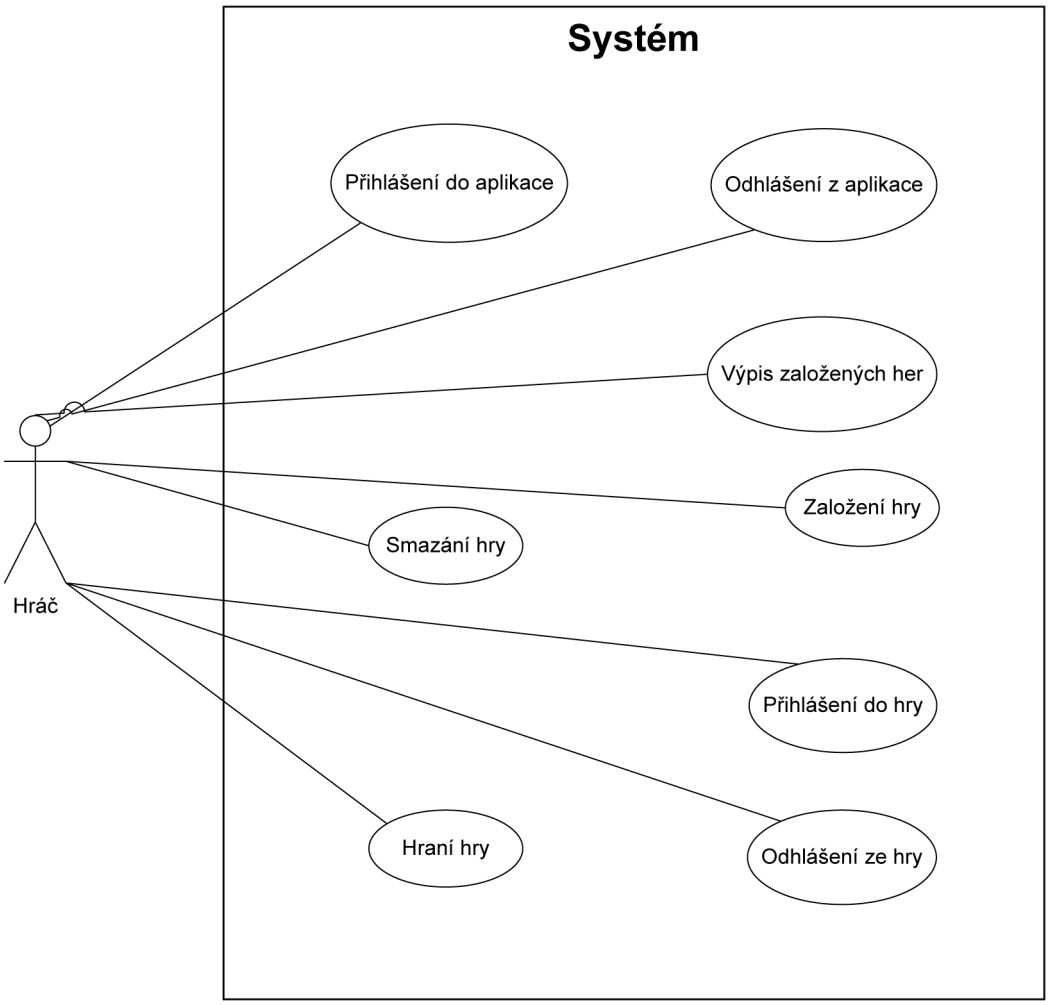
Tyto funkce jsou zachyceny diagramem případu užití na obrázku 2.

3.2 Datová analýza

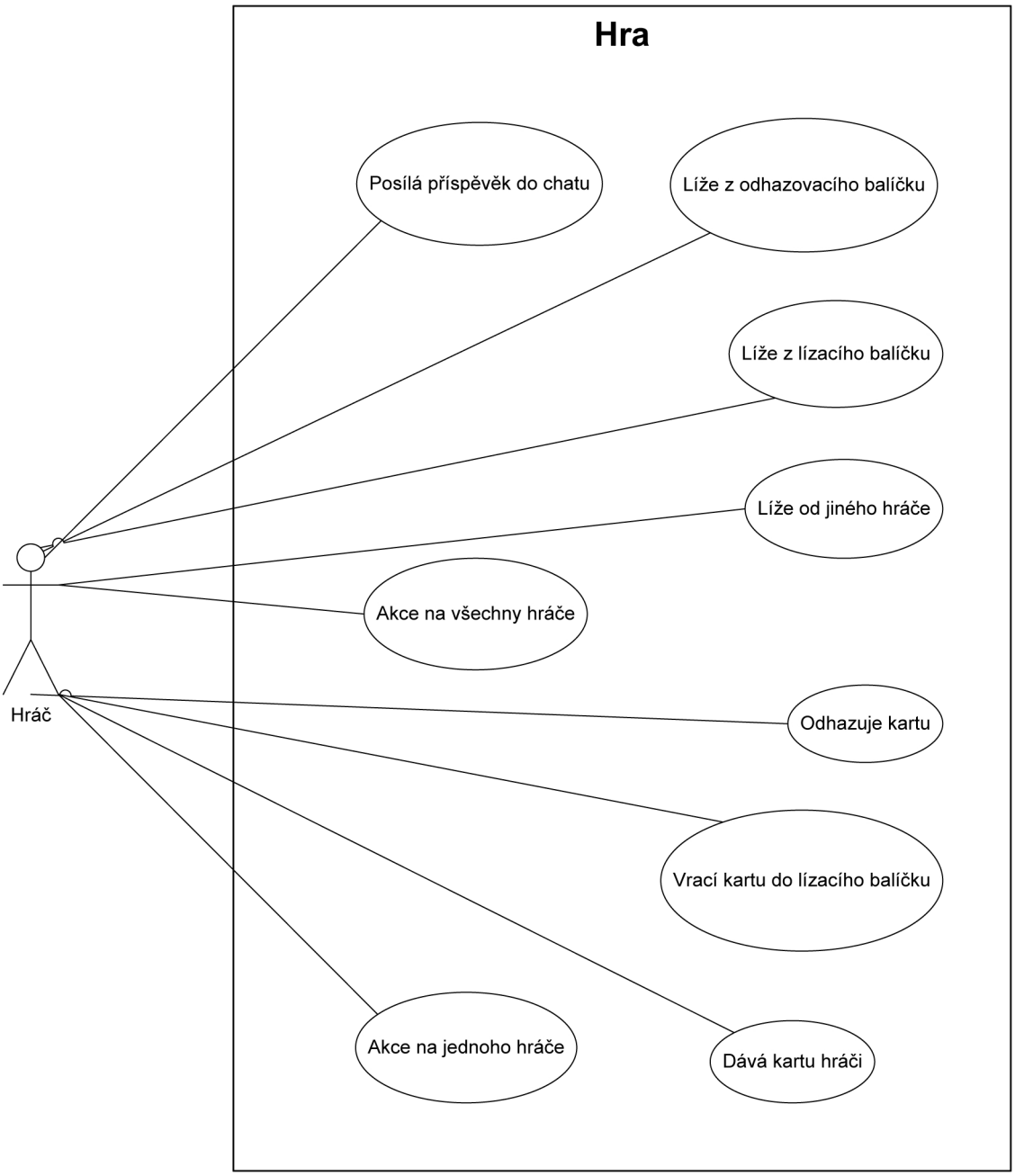
Důležitou částí návrhu systému je volba vhodného uložení dat, definice tabulek a jejich položek s definováním vazebních vztahů mezi nimi. Vytvoření jejich návrhu se označuje pojmem datová analýza.

Vstupy:

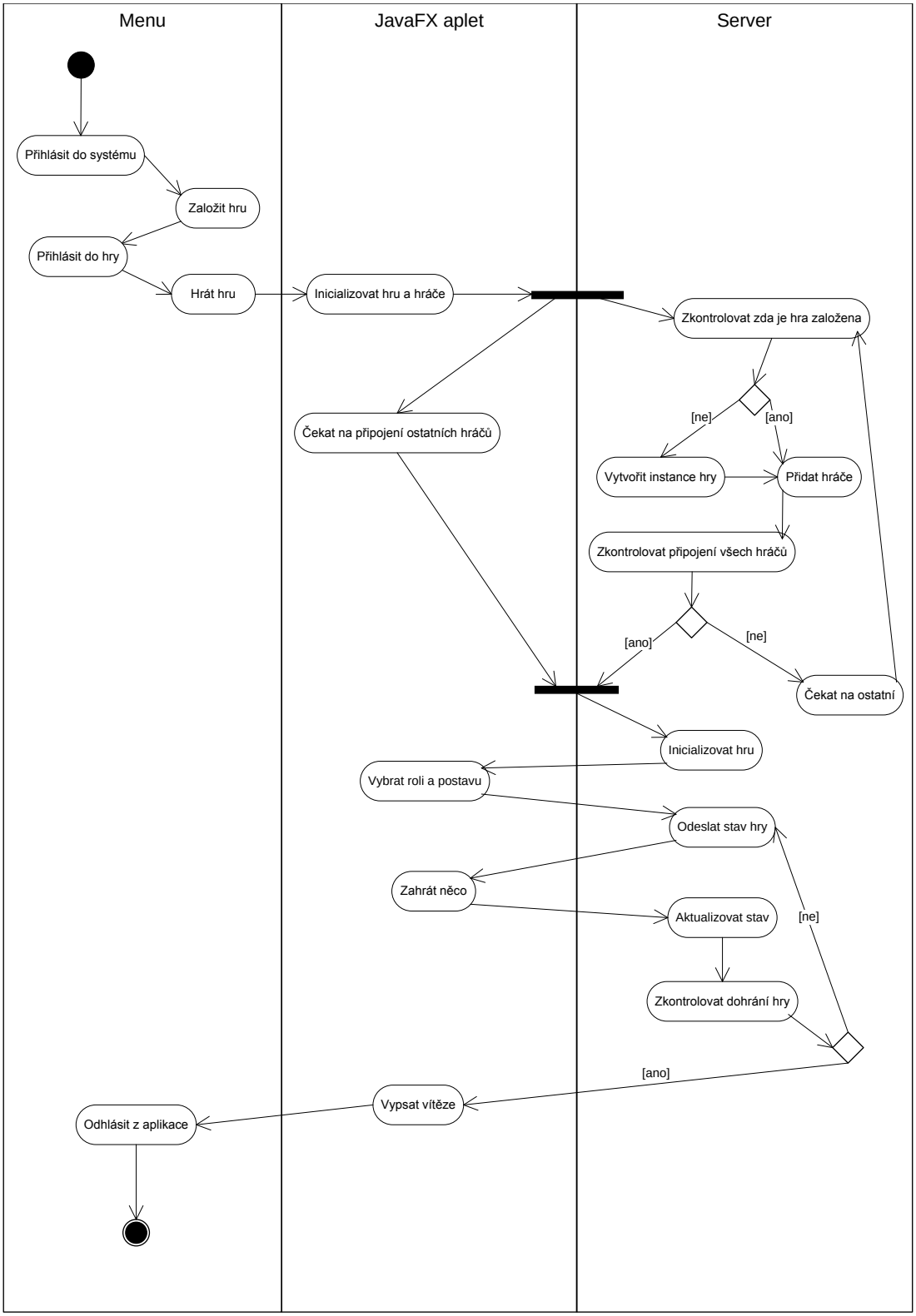
U hráče bude evidováno jeho identifikační číslo, přezdívka, role a postava, za kterou hraje, a karty v ruce a na stole.



Obrázek 1: Use case diagram systému



Obrázek 2: Use case diagram hry



Obrázek 3: Diagram aktivit

U hry bude evidováno její identifikační číslo, název, záznam, zda budou kontrolovány pravidla, datum a čas založení a hráči, kteří jsou do ní přihlášeni.

U hrací karty bude evidováno její indentifikační číslo, název karty v originále, český název, barva rámečku, označení a text na ní.

U postavy bude evidováno její identifikační číslo, jméno, vlastnost, kterou má, a počet životů.

U role bude evidováno její identifikační číslo, název a popis jejího úkolu.

Výstupy a funkce:

- registrace hráče
- registrace hry
- smazání hry
- přihlášení do hry
- odhlášení ze hry
- výpis založených her
- výpis hráčů ve hře
- hraní hry

3.2.1 Konceptuální schéma

Konceptuální schéma popisuje fakta o reálném světě, která se buď nemění vůbec nebo jen málo. Jeho úkolem je odstranit náhlý přechod od zájmové reality přímo k logickému schématu báze dat. Popisuje objekty zájmové reality a jejich vztahy mezi sebou. Pro vyjádření konceptuálního schématu slouží tzv. konceptuální modely, mezi které patří např. entitně-relační model.

Lineární zápis

V této části se nachází lineární zápis seznamu typů entit s jejich atributy. Název tabulky je zvýrazněn **tučně**, primární klíče podtržením a cizí klíče *kurzívou*.

Hra(hraId, nazev, kontrolovatPravidla, datum, cas, *hracId*)

Hrac(hracId, nick, *kartaId*, *postavaId*, *roleId*)

HraciKarta(kartaId, jmeno, ceskeJmeno, barva, oznaceni, text)

Postava(postavaId, jmeno, text, zivoty)

Role(roleId, nazev, popis)

Datový slovník

Tabulky číslo 1 a 2 ukazují příklad datového slovníku. Úplný datový slovník všech tabulek se nachází v příloze.

Hra

Tabulka číslo 1 znázorňuje tabulku, do které se ukládají hry. Je identifikována jedinečným číslem, který je automaticky generován. Při přidání záznamu je třeba uvést název hry, případně zapnout kontrolu pravidel. Ostatní atributy se nastaví automaticky.

Název	Datový typ	Klíč	NULL	Index	Popis
hraId	int(11)	ano	ne	ano	primární klíč
nazev	char(50)	ne	ano	ne	název hry
kontrolovatPravidla	tynyint(1)	ne	ano	ne	kontrola zapnuta ano/ne
datum	date	ne	ano	ne	datum založení hry
cas	time	ne	ano	ne	čas založení hry
hracId	int	ne	ano	ne	ID hráče ve hře

Tabulka 1: Datový slovník tabulky Hra

Postava

Tabulka číslo 2 znázorňuje tabulku, ve které jsou uložené dostupné postavy. Obsah této tabulky je neměnný. Slouží pouze pro uložení informací o postavách pro hru.

Název	Datový typ	Klíč	NULL	Index	Popis
postavaId	int(11)	ano	ne	ano	primární klíč
jmeno	char(50)	ne	ano	ne	jméno postavy
text	char(200)	ne	ano	ne	vlastnost postavy
zivoty	int(11)	ne	ano	ne	počet životů

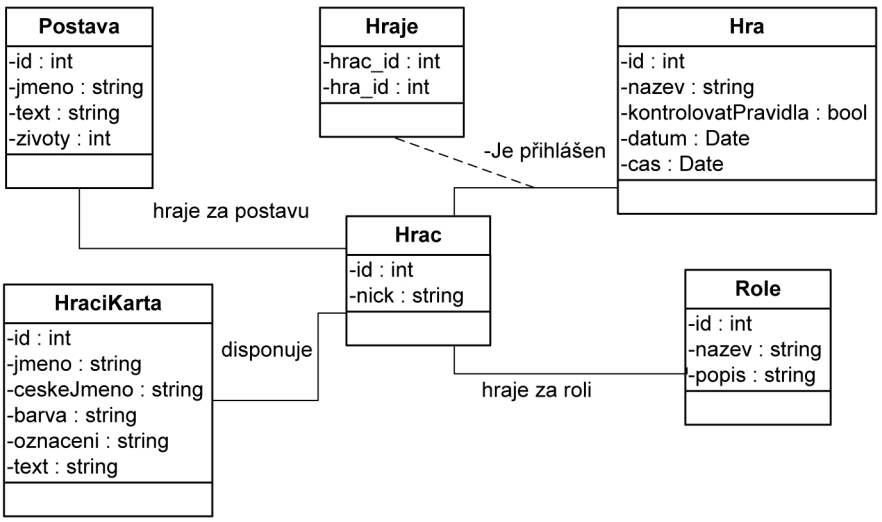
Tabulka 2: Datový slovník tabulky Postava

3.2.2 Class diagram - diagram tříd

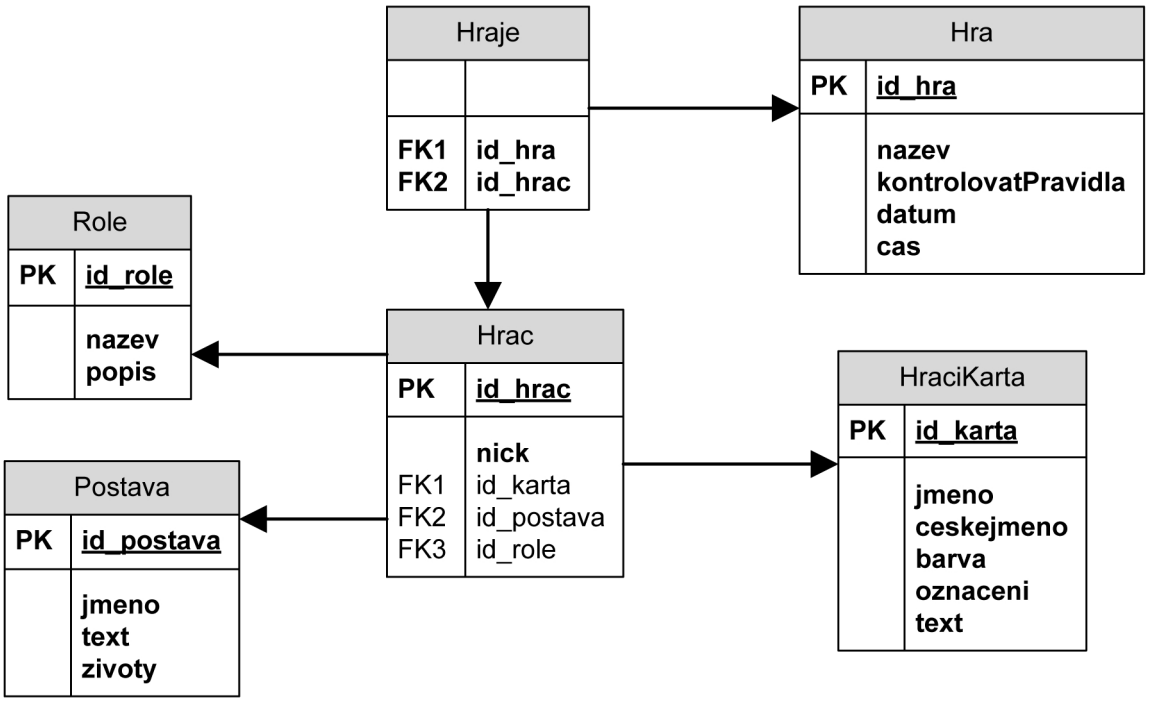
Diagram tříd zobrazuje statický pohled na systém, tj. zejména třídy (class), jako typy objektů, obsah tříd a statické vztahy, které mezi nimi existují [4]. Třídní diagram systému je na obrázku 4.

3.2.3 ER diagram

Entity relationship (ER) diagram slouží k modelování entitních typů a vazeb databáze. ER diagram systému se nachází na obrázku 5.



Obrázek 4: Třídní diagram



Obrázek 5: ER diagram

4 Návrh a implementace

V první části této kapitoly jsou uvedeny technologie, které byly k implementaci aplikace použity. Technologie, které jsou známé již delší dobu, jsou zde uvedeny spíše pro úplnost. Naopak technologiím, které zatím příliš rozšířeny nejsou, je věnována větší pozornost.

Samotná implementace aplikace je rozdělena na několik částí. Server zpracovávající logiku hry, webové uživatelské rozhraní pro menu, databázi a tenkého klienta aplikace (JavaFX aplikace). Princip tenkého klienta spočívá v jednoduchosti. Nedisponuje logikou hry, ta je přesunuta na server, se kterým klient komunikuje. Těmto částem se věnuji ve druhé části kapitoly. Postupně jsou rozepsány a vysvětleny s ukázkami zdrojových kódů a snímků.

4.1 Použité technologie

4.1.1 JSF

JSF je technologie pro vývoj webových aplikací. Byla vyvinuta společností Sun Microsystems a je součástí Java EE.

Rozděluje webovou aplikaci na dvě části. Část uživatelského rozhraní (GUI) popsaného pomocí speciálních XML tagů a část aplikační logiky v jazyce Java. Výměna dat mezi těmito částmi probíhá pomocí faces. Jedná se o soubory speciálních XML značek, kterým se specifikují odkazy na třídy aplikační logiky uložených na serveru.

V následujícím popisu mluvím o JSF ve verzi 2.0, kterou jsem použil v mé práci. V této verzi je doporučeno vytvářet stránky ve formátu XHTML. Nabízí podporu obrázků, CSS nebo JavasSriptu. Také byla přidána podpora událostí a byla rozšířena sada komponent.

GUI - Uživatelské rozhraní

Úkolem uživatelského rozhraní je uživateli prezentovat data, která mu zprostředkují faces.

Při psaní faces jsou využívány speciální XML značky importované z tzv. Tag Library Description (TLD) souborů. Tyto TLD soubory jsou součástí Java Server Faces knihoven. Tvůrce technologie (Sun Microsystems) vytvořil specifikace několika knihoven pro základní použití. Mezi ně patří knihovna tagů pro výpis textu, vstup textu, combo box atd.

Během implementace aplikace je možné používat více knihoven současně. Může ale nastat případ, kdy se v různých knihovnách vyskytují tagy se stejnými jmény. Tyto konflikty je potřeba vyřešit. K odlišení tagů, které mají stejný název, se používá prefix, kterým si programátor pojmenuje importovanou knihovnu.

Aplikační logika

Aplikační logiku obstarávají JavaBeany. Zde se již programátor nezajímá o to, jakým způsobem budou data prezentována uživateli. Beany běží na serveru, nikoliv u uživatele.

Z toho plyne, že uživateli stačí mít pouze HTML prohlížeč, nemusí však mít nainstalované JRE.

JavaBean je třída v programovacím jazyce Java, která splňuje následující podmínku. Třída je beanou tehdy, pokud má prázdný konstuktor. Dále platí, že k vlastnostem třídy se přistupuje pouze za použití get a set metod, případně is metod.

Managed beana je typ JavaBeany, která může přetrvávat v rámci dotazu, session nebo aplikace.

4.1.2 JavaFX

JavaFX je nová softwarová platforma na bázi platformy Java z dílny Sun Microsystems. Technologii začala vyvíjet společnost SeeBeyond, kterou později koupil Sun, kterého později koupil Oracle. Její využití je v oblasti vývoje tzv. RIA aplikacích zajišťujících interaktivitu webových stránek. K tomu používají animace, efekty, zvuky a videa. Technologie je konkurencí pro Adobe Flash, Adobe Flex, Adobe AIR, Microsoft Silverlight nebo OpenLaszlo.

Její první verze (JavaFX 1.0) vyšla v prosinci 2008 a současná verze nese označení JavaFX 1.3.

Spuštění JavaFX aplikací je možné více způsoby:

- **Standardní spuštění** - jako ostatní běžné aplikace.
- **Spuštění ve webovém prohlížeči** - aplikace se jeví jako aplet.
- **Java Web Start** - aplikace je stažena a následně spuštěna pomocí technologie Web Start.
- **Mobilní platforma** - spuštění na mobilních přístrojích.
- **TV platforma** - možnost spuštění aplikace na set-top box zařízeních, k ovládání slouží dálkový ovladač.

JavaFX Script

JavaFX technologie využívá k psaní kódu svůj specializovaný jazyk s názvem JavaFX Script (původně F3). Jedná se o skriptovací jazyk s kompilací, při které jsou programátorovi oznamovány syntaktické a jiné chyby.

JavaFX Script je staticky typovaný deklarativní skriptovací jazyk vycházející z platformy Java. Udává se, že jeho kód je oproti kódu Javy až 5x úspornější. Syntaxí může připomínat JavaScript. Jeho výhodou je poskytnutí automatického data bindingu a plné podpory 2D grafiky, swingovských komponent a deklarativních animací. Data binding je technika spojující logiku aplikace s uživatelským rozhraním. Při změně hodnoty dat se jejich změna projeví na elementech, které jsou s nimi svázány a naopak. Odpadá tedy nutnost použití listenerů jako je tomu v Javě. Umožňuje použití a práci s Java objekty.

Vývoj v JavaFX

Pro tvorbu aplikací v JavaFX je zapotřebí JavaFX SDK (současná verze JavaFX 1.3.1 SDK) a vývojové prostředí, kterým může být například Netbeans v poslední verzi 6.9.1 nebo Eclipse 3.5 Galileo.

4.1.3 Ostatní technologie

MySQL

MySQL je multiplatformní databázový systém, který vlastní společnost Oracle Corporation. Jeho výhodou je dostupnost pod bezplatnou licencí GPL. Díky čemuž je v současnosti velice rozšířen pro použití při tvorbě webových stránek. Využívá jazyka SQL.

Java

Populární objektově orientovaný programovací jazyk vyvinutý firmou Sun Microsystems.

XHTML

Značkovací jazyk určený k tvorbě hypertextových dokumentů vyvinutý standardizační organizací W3C.

CSS

Jazyk určený k popisu způsobu zobrazení stránek, rovněž vyvinutý standardizační organizací W3C.

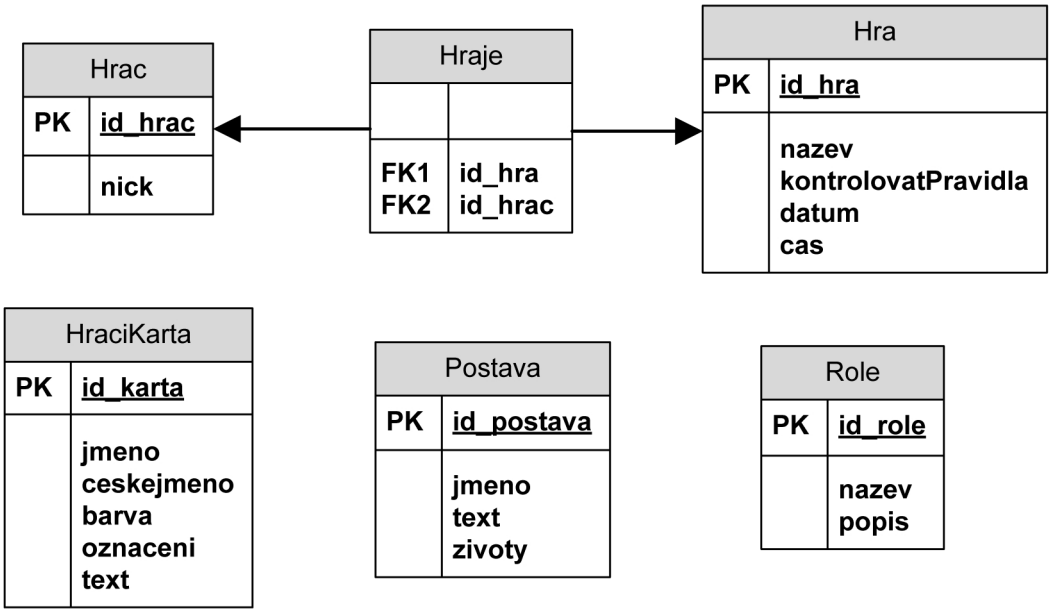
JSON

JavaScript Object Notation (JavaScriptový objektový zápis, JSON) je způsob zápisu dat (datový formát) nezávislý na počítačové platformě, určený pro přenos dat, která mohou být organizována v polích nebo agregována v objektech. Vstupem je libovolná datová struktura (číslo, řetězec, boolean, objekt nebo z nich složené pole) a výstupem je vždy řetězec.

Na výpisu 1 je příklad JSON. Řetězec je výsledkem převodu objektu typu Hrac, který má atributy nick a id. Jak je vidět, prvky v JSON jsou dvojice oddělené dvojtečkou, ve kterých je nejprve uvedena proměnná a následně její hodnota.

```
{"Hrac": {"nick": "Honza", "id": 9}}
```

Výpis 1: Příklad JSON



Obrázek 6: Konečný ER diagram

4.2 Databáze

K implementaci databáze jsem použil technologii MySQL. Databáze slouží k dvěma účelům, které budou popsány dále.

V aplikaci je databáze použita pro uložení informací o:

- hráči,
- hře,
- kdo hraje jakou hru,
- postavě,
- roli,
- hrací kartě.

Pro účely implementace byl návrh databáze upraven, konečná podoba ER diagramu je na obrázku 6.

Diagram ukazuje jedinou vazbu, a to mezi hráčem a hrou. Tyto tabulky jsou jediné, jejichž obsah se mění. Obsah zbylých tabulek je totiž neměnný a jsou tedy využívány pouze pro načítání informací. Týká se to tabulek pro ukládání karet, rolí a postav.

Příklad vytvoření tabulky znázorňuje výpis 2, který popisuje vytvoření tabulky Postava. Zdrojové kódy pro vytvoření všech tabulek jsou uvedeny v příloze.

```
CREATE TABLE postava
(
    id  int  NOT NULL PRIMARY KEY auto_increment,
    jmeno char(50) CHARACTER SET utf8 COLLATE utf8_czech_ci ,
    text  char(200) CHARACTER SET utf8 COLLATE utf8_czech_ci ,
    zivoty  int
) DEFAULT CHARACTER SET utf8 COLLATE utf8_general_ci;
```

Výpis 2: Ukázka vytvoření tabulky Postava

4.3 Webové rozhraní

Webové rozhraní bylo implementováno pomocí technologií JSF 2.0, XHTML a CSS. Plní funkci menu celého systému a je do něj umístěn JavaFX aplet, ve kterém probíhá samotná hra.

Funkce webového systému byly popsány výše. Následuje popis jejich implementace. Správné přihlášení je zajištěno kontrolou přezdívky. Uživatel musí zadat alespoň jeden znak, v opačném případě je systémem odmítnut. Pro zakládání her slouží k tomu určená stránka, stejně tak pro výpis seznamu her a čekání na přihlášení ostatních hráčů do hry.

Aplikační logika webu obsahuje klienta pro spojení a práci s databází. Při založení hry se údaj o této hře uloží do databáze. Toto se děje také při přihlášení hráče do hry. Nejprve se uloží uživatel do tabulky Hrac, a následně se vloží záznam, že hráč hraje hru. Vypsání založených her probíhá tak, že klient zjistí z databáze všechny založené hry, které předá aplikační logice, a ta je na stránce zobrazí v tabulce. V této tabulce jsou rovněž tlačítka pro přihlášení do hry nebo její smazání.

Při požadavku na spuštění hry je vygenerována stránka s apletem. Aby mohl aplet správně vykonávat svojí funkci, potřebuje při jeho spuštění dva parametry. Jedná se o ID hráče a ID hry, kterou hráč hraje. Na výpisu 3 je uveden příklad vygenerovaného skriptu pro spuštění JavaFX apletu. Je zde vidět předání parametrů apletu, první parametr značí ID hráče a druhý ID hry.

Následují snímky obrazovek z aplikace. Přihlášení uživatele na obrázku 7, zakládání hry na obrázku 8, seznam založených her na obrázku 9 a nakonec, na obrázku 10 stránka pro vyčkávání na přihlášení ostatních hráčů. V příloze se nachází snímek z rozehrané hry.

```
<script src="http :// dl.javafx.com/1.3/dtfx.js"></script>
<script>
    javafx(
        {
            archive: "BangFX.jar",
            draggable: true,
            width: 1200,
            height: 550,
            code: "bang.Main",
            name: "BangFX"
        },
  {
    hID: 13,
    hraID: 9
  }
   );
</script>
```

Výpis 3: Příklad vygenerovaného skriptu pro spuštění JavaFX apletu

4.4 Server

Server aplikace je napsán v programovacím jazyce Java a obsahuje veškerou logiku hry. Má za úkol zprostředkovávat komunikaci mezi hráči ve hře, hlídání pravidel a upravování stavu databáze. Server zvládá obsluhovat současně více klientů (hráčů) a ke komunikaci s nimi používá TCP sockety.

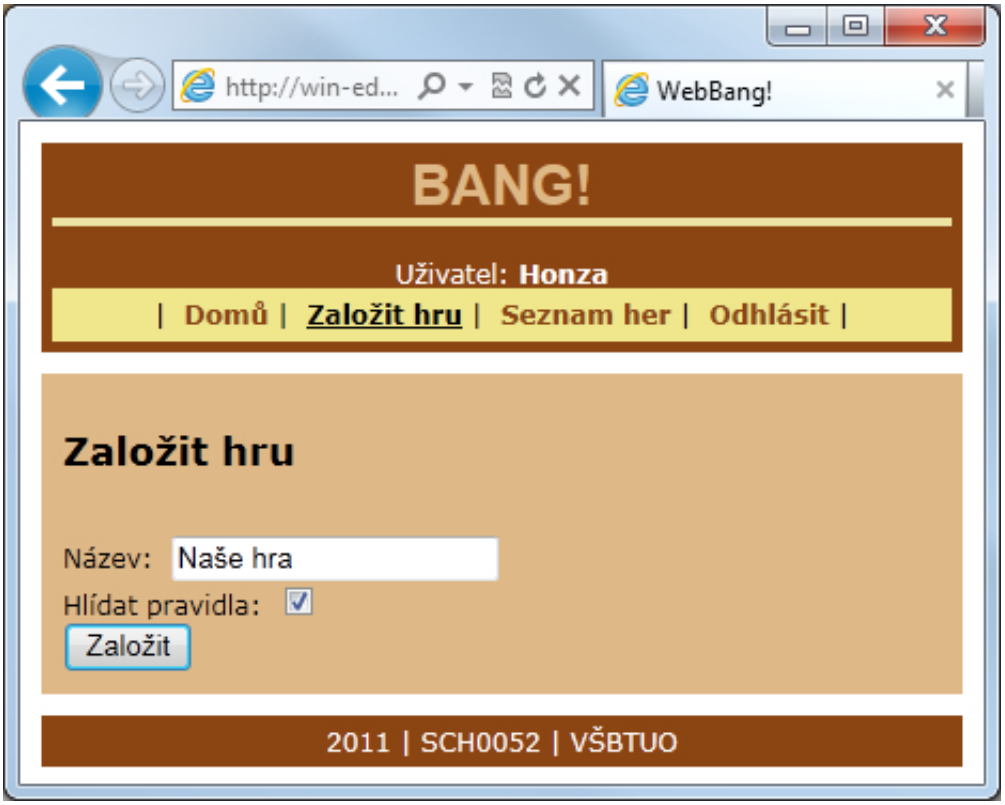
Hlavní části serveru:

- Třída DbManager, která má na starosti práci s databází.
- Třída Klient, která slouží pro obsluhu připojených klientů. Zde se nachází komunikace s pomocí socketů.
- Třída HraLogika, která obsahuje logiku hry. Udržuje v sobě hráče a jejich stav. Provádí veškeré operace jako lízání, ubrání životu atd.
- Třída Pravidla. Jedná se o statickou třídu, která slouží pro kontrolu pravidel.

4.4.1 Činnost serveru

Server při spouštění nejprve zjistí, zda má spojení s databází. Pokud toto spojení nemá, nemůže plnit svůj účel, proto nedostupnost databáze oznámí a ukončí se. Při úspěšném navázání spojení toto spojení zruší a začne poslouchat na předem nastaveném portu, kde vyčkává na připojení klientů. V případě navázání spojení s klientem, předá spojení třídě Klient, která vykonává jeho obsluhu, poté tuto obsluhu spustí v dalším vlákně a čeká dále na připojení dalších klientů.

Obrázek 7: Stránka pro přihlášení uživatele do systému



Obrázek 8: Zakládání hry

Obrázek 9: Seznam všech založených her

Obrázek 10: Čekání na ostatní hráče

4.4.2 Obsluha klienta

Třída Klient pracuje v nekonečné smyčce, ve které se vykonávají následující kroky:

1. Příjem dat od uživatele (hráče),
2. dekódování dat,
3. vyhodnocení dat.

Při navázání spojení vzdálený klient nejprve odešle požadavek na inicializaci. V ní uvede svoji přezdívku ve hře a identifikační číslo. Server poté zjistí, zda je hra s tímo ID založena a v případě že ne, zjistí z databáze počet přihlášených hráčů do ní. Dále načte všechny hrací karty a karty rolí a postav. Nakonec z databáze smaže hráče a hru, aby se neobjevovala v nabídce založených her v menu. Po přihlášení dalších hráčů pouze porovnává počet přihlášených hráčů do hry s počtem již připojených hráčů, a pokud jsou si rovny (jsou připojeni všichni přihlášení hráči), odešle na výběr karty rolí a postav.

Dekódování přijatých dat

Data, která jsou posílána mezi uživateli a serverem, jsou ve formě JSON. Tato forma neumožňuje definovat znakovou sadu přenášeného obsahu, proto je potřeba druhý krok.

Z takto upraveného JSON řetězce je nyní možné vytvořit nový objekt třídy Pozadavek, který obsahuje ID o jaký požadavek se jedná, dále dvě proměnné pro celá čísla a nakonec řetězec znaků. Objekt obsahuje vždy pouze data potřebná k operaci, jediným povinným atributem je tedy ID požadavku.

Vyhodnocení

Vyhodnocení dat probíhá v následujících krocích:

1. Zjištění ID požadavku. ID požadavku je prezentováno výčtovou hodnotou, jejíž název napovídá metodu, která bude pro obsloužení vykonána.
2. Načtení dat (parametrů), se kterými bude volána metoda.
3. Vykonání požadavku.
4. Poslání klientům aktuální stav hry.

Vykonání požadavku

Třída Klient uchovává veškeré hry ve statické proměnné typu Hastable. Klíčem hry je její identifikační číslo. Při vykonání metody se musí nejprve vybrat požadovaná hra. Po výběru se v této hře zavolá metoda.

```
private void broadcast(PozadavekID co) {

  for (Klient k : getKlientiVeHre(this.idHry)) {
    switch (co) {
      case HRACI:
        k.odeslat(PozadavekID.HRACI.toString());
        k.odeslat(getHraci());
        break;
      case KARTA_ODHAZOVACI_BALIK:
        k.odeslat(PozadavekID.KARTA_ODHAZOVACI_BALIK.toString());
        k.odeslat(getOdhazovaciVrchniKarta());
        break;

        // ...

    } //switch
  } // for
} //broadcast
```

Výpis 4: Část broadcastové metody

Poslání aktuálního stavu

Po vykonání operace ve hře je nutné tuto změnu zobrazit u všech hráčů. Třída Klient obsahuje statický seznam všech hráčů, kteří jsou momentálně k serveru připojeni. Část metody pro posílání aktuálního stavu hry je na výpisu 4. Metoda má jako vstupní parametr předán název požadavku, resp. odpovědi, kterou bude klientům posílat. Metoda getKlientiVeHre vrací hráče v dané hře, které vybere ze seznamu všech připojených klientů (každý klient obsahuje informaci jakou hru hraje). Tyto hráče následně prochází a zasílá jim nejprve název požadavku, aby klient na druhé straně věděl, jaká data mu přijdou, a poté samotná data.

4.4.3 Reakce na nečekané události

Server reaguje na nečekané události, jako je například ztráta spojení s některým z hráčů. Pokud nastane taková událost, server zjistí, které hry byl hráč členem, a všem ostatním v dané hře odešla informaci o události a o konci hry.

4.4.4 Pravidla

Ke kontrole pravidel jsem vytvořil speciální třídu. Tato třída obsahuje statické metody k ověření přípustnosti dané operace. Na výpisu 5 je část jedné metody. Jedná se o metodu ke kontrole při vykládání karty. Metoda postupně zjišťuje splněné podmínky. Nejprve ověří, že je karta modrá, poté zjišťuje, zda je to karta zbraně a pokud ano, ověří, že jiná zbraň ještě vyložena nebyla. Takto postupuje dále, až ověří všechny podmínky a vrací boolean hodnotu, zda se tah smí provést nebo ne. Výpis také ukazuje, jaké parametry metoda potřebuje. Jsou to identifikační číslo karty a hráče, a také seznam všech hráčů.

```
public static boolean muzeVylozitKartu(int hID, int idKarty, Map<Integer, Hrac> hraci) {
    boolean vrat = true;
    if (idKarty > 17) {
        //karta s cislem 18 a vys neni modra, nevyklada se
        vrat = false;
    } else if (kartaJeZbran(idKarty)) {
        if (hracMaVylozenouZbran(hraci.get(hID))) {
            //na stole ma zbran a snazi se vylozit druhou
            vrat = false;
        }

    // ...

    return vrat;
}//muzeVylozitKartu
```

Výpis 5: Část metody pro kontrolu vykládání karty

Pravidla, které jsem implementoval, nejsou úplně všechna. Bylo to z toho důvodu, že se jedná o hru a hráči by při ní měli udržovat pozornost. V případě, že by se kontrolovala každá událost, by hra ztrácela na své zábavnosti. Takto se její online verze částečně přiblížila realitě. V příloze se nacházejí kompletní pravidla ke hře. Pravidla, která se při hře kontrolují, jsou následující:

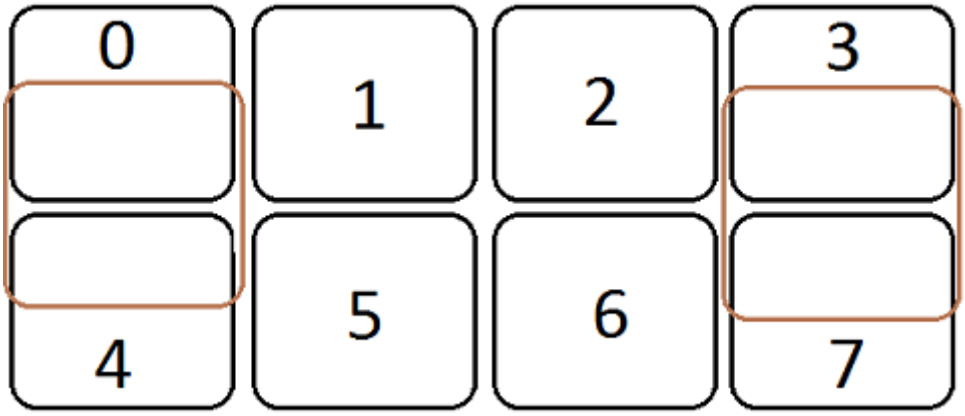
- Hru hraje 4 až 7 hráčů. Tato kontrola se provádí již na úrovni webového rozhraní a pro účely testování byla vypnuta.
- Hráč vidí na hráče. Pro potřeby karty BANG!.
- Hráč může ukončit tah. Ověří se počet jeho karet.
- Hráč může vyložit kartu před sebe na stůl.
- Hráč může dát kartu jinému hráči.
- Hra končí. Po zastřelení hráče proběhne kontrola, zda jeho smrt znamená konec hry. Pokud ano, metoda vrací výsledek hry.

4.4.5 Změny oproti návrhu

Změny návrhu se týkají třídy Hrac. Není nutné, aby každý hráč udržovl celé objekty karet. Při posílání stavu hry ze serveru klientům by kvůli velkému množství dat mohlo dojít ke zdržování. Proto se uchovávají pouze identifikační čísla karet.

4.5 JavaFX aplikace

JavaFX aplikace tvoří grafické rozhraní hry. Komunikuje se serverem, kterému posílá požadavky na vykonání operace a naopak od něj přijímá aktuální stav hry a zobrazuje jej hráči. Obsahuje také nápovědu jak hru hrát.



Obrázek 11: Rozvržení komponent

Vytvoření aplikace si vyžádalo řadu vlastních komponent. Základní komponentou je Kontejner, který plní funkci stolu, na němž jsou umístěné komponenty pro hráče, karty, balíčky, nápovědu a hokynářství. Nejdůležitější komponenty jsou popsány detailněji.

4.5.1 Grafická část

Komponenta s balíčky v sobě obsahuje také chat, historii tahů, kartu se symboly a místo pro zobrazení náhledu karty, nad kterou se nachází kurzor myši. Druhá důležitá komponenta slouží pro zobrazení hráče. Obsahuje veškeré informace o něm. Při návrhu grafického prostředí jsem musel počítat s tím, že je potřeba zobrazit až 7 hráčů a k tomu balíčky i na menších rozlišení. Na obrázku 11 je zobrazen návrh rozmístění komponent. Pozice s číslem 0 slouží pro komponentu s balíčky a pozice 1 až 7 pro hráče. V případě, že budou hrát pouze 4 hráči, se posune komponenta s balíčky na hnědé pole vlevo. Pokud bude hrát 5 hráčů, posune se navíc hráč na pozici 3 do druhého hnědého pole.

Komponenta Karta

Komponenta karty řeší práci s kartou. Při kliknutí na kartu je nutné toto dát viditelně najevo. Z toho důvodu komponenta reaguje zmenšením obrázku karty a zobrazení červené značky.

Každé kartě byla přidána možnost měnit svojí vertikální polohu. Pro karty v ruce směrem nahoru (pro vyložení na stůl) a naopak. Na výpisu 6 je část třídy pro posun karty směrem dolů. Po stisku tlačítka myši se uloží pozice kurzoru. Při jeho pohybu se kontroluje směr, a pokud se pohybuje směrem dolů, mění se také souřadnice umístění karty. Třída pro tah kartou dolů funguje na stejném principu.

```
init {
    target.onMousePressed = function(e: MouseEvent): Void {
                startX = e.sceneX - target.translateX;
                startY = e.sceneY - target.translateY;
            }
    target.onMouseDragged = function(e: MouseEvent): Void {
                var ty = e.sceneY - startY;
                if (ty < 0) { ty = 0; }
                if (ty > maxY) { ty = maxY; }
                target.translateY = ty;
            }
}
```

Výpis 6: Část třídy TahDolu pro posun karty směrem dolů

Komponenta Hrac

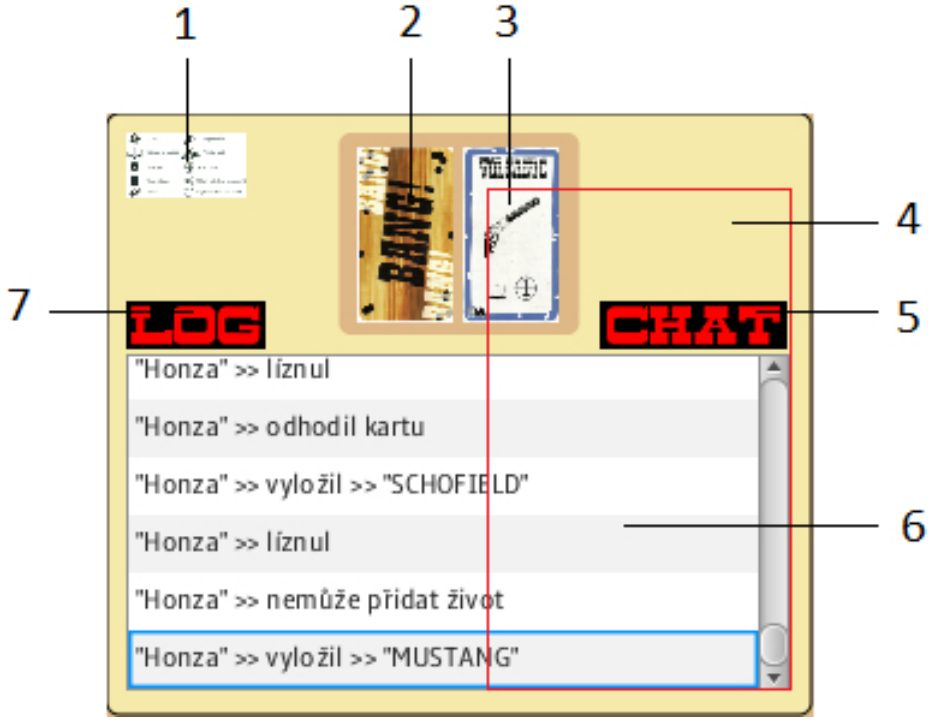
Komponenta hráče zobrazuje jeho stav. Úprava životů je možná pomocí tlačítek. Pokud chce hráč vyložit kartu na stůl, učiní tak tažením karty nahoru. Pokud ji chce vrátit do ruky, táhne směrem dolů. Pro označení karty, kterou chce hrát, na ní klikne levým tlačítkem myši (karta se červeně označí), a poté klikne na tlačítko revoleru hráče, na kterého hraje. V případě, že hraje kartu na všechny, klikne na kohokoli. Pokud se rozmyslí a kartu hrát nechce, klikne na ní pravým tlačítkem myši. Pro odhození karty na ní stačí dvakrát poklepat. Svůj tah hráč ukončí kliknutím na tlačítko EXIT. Podoba komponenty je na obrázku 12. Přehled prvků:

- 1 - Přezdívka hráče.
- 2 - Karta role. Šerif je viditelný všem.
- 3 - Tlačítko EXIT. Slouží pro ukončení tahu. U ostatních hráčů je na jeho místě tlačítko s revoleverem, které slouží pro určení, že karta je hrána na něj.
- 4 - Vrchní karty, které jsou na stole a jsou vidět všem hráčům.
- 5 - Spodní karty, které drží hráč v ruce a vidí je pouze on.
- 6 - Tlačítka pro přidání a ubrání životů.
- 7 - Karta postavy. Tato karta se posunuje po vertikální ose v závislosti na počtu nábojů (životů).
- 8 - Obrácená karta s počtem nábojů (životů).

Komponenta Balicky

Komponenta s balíčky obsahuje také chat pro komunikaci hráčů, historii tahů hry a kartu se symboly, která se po najetí myši zvětší. Líznutí karty proběhne kliknutím na

Obrázek 12: Komponenta hráče



Obrázek 13: Komponenta balíčků

požadovaný balík, ze kterého hráč líže. Pro odeslání příspěvku do chatu stačí stisknout klávesu Enter. Podoba komponenty je na obrázku 13. Přehled prvků:

- 1 - Karta symbolů.
- 2 - Lízací balík.
- 3 - Balík s odhozenými kartami.
- 4 - Místo, kde se zobrazuje náhled karet ve větší velikosti.
- 5 - Tlačítko pro zobrazení chatu a psaní zpráv.
- 6 - Historie hry / Chat.
- 7 - Tlačítko pro zobrazení historie hry.

4.5.2 Logická část

Logickou část tvoří nastavení, práce klienta pro připojení k serveru a prezentace přijatých dat hráči. JavaFX aplikace také musí reagovat na přerušení, či dokonce nenavázání spojení se serverem.

Nastavení grafického prostředí je uloženo v konfiguračním souboru Kfg.fx. Jsou zde definovány velikosti komponent a jejich rozmístění a barvy. Také se zde nacházejí globální proměnné, které musí být přístupné z více částí JavaFX aplikace, např. obrázek karty náhledu, popisek karet rolí a postav pro lepší čitelnost nebo klient pro komunikaci se serverem.

Informace o hráčích jsou uloženy v jejich modelech, a z nich jsou pak načítány do grafické komponenty. Při přijetí nových modelů se provede jejich aktualizace a také aktualizace komponent.

Přijatá data ze serveru jsou ve formě JSON. K jejich převodu do modelů hráčů byla použita knihovna json-lib-2.4, která umožňuje převod přímo do objektu.

Klient

JavaFX (ve verzi 1.3) technologie nepodporuje vícevláknové aplikace, z toho důvodu musel být klient napsán v jazyce Java. Je totiž důležité jedním vláknem data odesílat a druhým přijímat. Pokud by toto nebylo umožněno, došlo by k zamrzání GUI aplikace.

Přenesení dat z javovského klienta do části JavaFX jsem řešil implementací rozhraní SocketListener v obou částech. Rozhraní obsahuje funkce, které budou volány při přijetí dat.

Při spuštění JavaFX aplikace se automaticky zavolá metoda pro připojení k serveru. V případě, že je spojení neúspěšné, klient tuto informaci předá kontejneru, a ten zobrazí hlášku. Pokud spojení proběhne úspěšně, vytvoří se instance třídy implementující rozhraní. Při přijetí dat se volá metoda z hlavního vlákna klienta. V této metodě se zavolá instance rozhraní, díky čemuž dojde k zavolání téže metody v kontejneru. Kontejner následně pracuje s daty, např. aktualizace gui nebo vypsání zprávy. Příklad metody je na výpisu 7. Jedná se o metodu pro obsluhu příjmu modelů hráčů.

```
@Override
public void hraci(final List<HracModel> hraci) {
    Entry.deferAction(new Runnable() {

        @Override
        public void run() {
            fxListener.hraci(hraci);
        }
    });
}
```

Výpis 7: Implementace metody z rozhraní SocketListener

4.5.3 Úprava pro nasazení apletu

Při nasazení apletu do webové stránky bylo nutné podepsat jak aplet samotný, tak veškeré knihovny třetích stran, které jsem použil. Pro každou knihovnu jsem vytvořil JNLP soubor, a ten jsem poté importoval v hlavním JNLP souboru pro celý jar soubor.

5 Testování

V této kapitole popisuji problémy a poznatky při nasazení a testování aplikace.

5.1 Nasazení

Celá aplikace byla nasazována ve třech částech.

První částí je databáze. K jejímu nasazení byl použit phpMyAdmin na serveru win456.vsb.cz a její vytvoření proběhlo pomocí jednoho inicializačního skriptu. Tento skript je součástí přílohy.

Druhou částí byla webová aplikace obsahující aplet s hrou. Tato část byla nahrána na školní server Glashfish. Nahrání proběhlo deployováním war souboru. Adresa, na které je možné aplikaci využívat, je http://win-edu.cs.vsb.cz:8080/WebBang/.

K nasazení třetí části, serveru obsahující logiku hry a komunikaci s hráči, byl použit opět server win-edu.cs.vsb.cz.

5.2 Testování

Testování probíhalo používáním aplikace. Při pozorování jejího chování se přišlo na několik poznatků a návrhů pro úpravu jak chování, tak i grafické stránky hry.

Tabulka 3 ukazuje prohlížeče, na kterých byla aplikace spuštěna a říká, zda byl její běh v pořádku.

Platforma	Prohlížeč	Verze	Běh aplikace
Windows	Opera	11.01	Funkční.
Windows	Opera	11.10	Funkční.
Windows	Chrome	10.0	Nefunkční.
Windows	Chrome	11.0	Nefunkční.
Windows	SeaMonkey	2.0.13	Funkční.
Windows	Firefox	4.0	Funkční.
Windows	Maxthon	3.0	Nefunkční.
Windows	Netscape Navigator	9.0.0.6	Nefunkční.
Windows	Avant Browser	2010, build 131	Funkční.
Windows	Internet Explorer	8	Funkční.
Windows	Internet Explorer	9	Funkční.
Windows	Safari	5.0.5	Nefunkční.
Linux	Opera	11.10	Funkční.
Linux	Firefox	4.0	Funkční.

Tabulka 3: Přehled prohlížečů

Během testování (a také před ním) se přišlo na následující problémy.

- Knihovna GSON. První problém se objevil již při spuštění apletu. Byl zapříčiněn použitím knihovny GSON od firmy Google. Tuto knihovnu jsem používal pro úpravu

přijatých dat ze serveru, kdy probíhal před z dat ve tvaru JSON na objekty. Tato knihovna využívá reflexe, která není povolená pro použití v apletu z důvodu bezpečnosti. Toto bylo odstraněno použitím knihovny JSON-lib.

- Problém s rychlostí. Protože jsem se snažil, aby aplikace byla oku příjemná, použil jsem grafické efekty. Přestože jich nebylo mnoho, jejich použitím se animace staly nepříjemně trhané a došlo také k celkovému zhoršení odezvy hry. Po odstranění efektů se rychlost zvýšila.
- Podepsání jar souborů. V části JavaFX aplikace jsem použil více knihoven třetích stran, které byly podepsány vlastními klíči. Jelikož výsledná JavaFX aplikace musela být také podepsána, došlo k chybě z důvodu podepsání různými klíči. Řešení spočívalo ve vytvoření jnlp souboru pro každou knihovnu a následně tyto jnlp soubory přidat do resources hlavního jnlp souboru.
- Codebase a HREF. Při načítání apletu na cizím počítači se objevila chyba nenalezení aplikace. Tuto chybu zapříčinila adresa na něj v jnlp souboru. Smázaním položek codebase a HREF byla chyba odstraněna.
- Spojení s MySQL. V původní implementaci serveru probíhálo připojení na databázi ihned při jeho spuštění. MySQL server však toto spojení po 8 hodinách zavřel a klient vyhodil výjimku MySQLNonTransientConnectionException. Z toho důvodu se nyní spojení s databází vytváří až při připojení klienta.
- Načtení kaskádových stylů. Při první návštěvě stránky pro přihlášení se nenačtou kaskádové styly. Po dalším přístupu již ano. Na příčinu této závady se mi přijít nepodařilo.

6 Závěr

Cílem mé bakalářské práce bylo seznámit se s karetní hrou BANG!, zorientovat se v pravidlech a implementovat ji pomocí Java EE 6. V práci bylo propojeno více technologií. Jedná se o databázi MySQL, webové rozhraní implementované pomocí JSF 2.0, server naprogramovaný v jazyce Java a aplet pro hraní v nové technologii JavaFX.

Funkční online verzi této hry se mi nepodařilo najít, proto se domnívám, že zatím implementována nebyla.

Kromě toho, že jsem se naučil společenskou hru, kterou jsem doposud neznal, mi práce přinesla mnoho praktických zkušeností a poznatků. Díky práci jsem naimplementoval aplikaci typu klient-server, poznal nové technologie, a především jsem je spojil dohromady v jeden funkční celek. Největším přínosem však bylo seznámení se zcela novou technologií JavaFX. Bohužel je vývoj této technologie nejistý.

6.1 Návrhy do budoucnosti

Budoucím přínosem pro aplikaci by bylo doimplementování některého rozšíření hry, která jsou dostupná. Patří do nich Dodge City, High Noon, Fistful, Face Off, Wild West Show. Tato rozšíření přináší například možnost hraní ve třech nebo osmi hráčích nebo použití nových karet.

Také by bylo možné pozměnit grafické prostředí a udělat jej dynamickým. V současné době je dáno pevně.

Do budoucna by bylo vhodné předělat implementaci kontroly pravidel. Pro kontrolu více pravidel je lepší implementovat inteligentnější mechanizmus.

Jan Schovánek

7 Reference

[1] ANDERSON, Gail; ANDERSON, Paul. *Essential JavaFX*, Anderson Software Group, May 2009.

[2] ŘÁBEK, Stanislav; BLAŽKO, Martin. *Bang!* [online]. c2011. Dostupné z: <http://www.bang.cz/>.

[3] Albi Česká Republika a.s. *Bang!* [online]. c2007, [cit.2011-4-28]. Dostupné z: <http://www.modernihry.cz/files/Bang.pdf>.

[4] PAVUS, M. *Class diagram - diagram tříd* [online]. c2005, [cit.2011-4-28]. Dostupné z: <http://mpavus.wz.cz/uml/uml-s-class-3-3-1.php>.

[5] REJNKOVÁ, Petra. *Příklady použití diagramů UML 2.0* [online]. c2009, [cit.2011-4-28]. Dostupné z: <http://uml.czweb.org/diagram_aktivit.htm>.

[6] JSON-lib. *JSON-lib* [online]. c2010. Dostupné z: <http://json-lib.sourceforge.net/index.html>.

[7] JELÍNEK, Lukáš. *Java (19) - síťová komunikace II* [online]. c2005. Dostupné z: <http://www.linuxsoft.cz/article.php?id_article=977>.

[8] JENDROCK, Eric; EVANS, Ian; GOLLAPUDI, Devika; HAASE, Kim; SRIVATHSA, Chinmayee. *The Java EE 6 Tutorial* [online]. c2011. Dostupné z: <http://download.oracle.com/javaee/6/tutorial/doc/>.

[9] Oracle. *Sample Gallery* [online]. c2010. Dostupné z: <http://javafx.com/samples/>.

[10] CONNORS, Jim. *JavaFX, Sockets and Threading: Lessons Learned* [online]. c2010. Dostupné z: <http://blogs.sun.com/jtc/entry/javafx_sockets_and_threading_lessons>.

[11] Oracle. *Building GUI Applications With JavaFX - Tutorial Overview* [online]. c2011. Dostupné z: <http://download.oracle.com/javafx/1.3/tutorials/ui/>.

A Obsah CD

Následující tabulka popisuje umístění souborů na CD a jejich popis.

Adresář	**Popis**
/doc/api	JavaDoc částí aplikace
/doc/thesis	Text bakalářské práce
/src	Zdrojové soubory
/scripts	Skripty pro vytvoření a naplnění tabulek

Tabulka 4: Obsah CD

B Snímky

Obrázek 14: Snímek průběhu hry

C Datový slovník

Na následujících tabulkách je úplný datový slovník.

Název	Datový typ	Klíč	NULL	Index	Popis
hraId	int(11)	ano	ne	ano	primární klíč
nazev	char(50)	ne	ano	ne	název hry
kontrolovatPravidla	tynyint(1)	ne	ano	ne	kontrola zapnuta ano/ne
datum	date	ne	ano	ne	datum založení hry
cas	time	ne	ano	ne	čas založení hry

Tabulka 5: Hra

Název	Datový typ	Klíč	NULL	Index	Popis
hracId	int(11)	ano	ne	ano	primární klíč
nick	char(50)	ne	ano	ne	přezdívka hráče

Tabulka 6: Hrac

Název	Datový typ	Klíč	NULL	Index	Popis
hracId	int(11)	ano	ne	ano	
hraId	int(11)	ano	ne	ano	cizí klíč

Tabulka 7: Hraje

Název	Datový typ	Klíč	NULL	Index	Popis
roleId	int(11)	ano	ne	ano	primární klíč
nazev	char(30)	ne	ano	ne	jméno role
popis	char(150)	ne	ano	ne	úloha role

Tabulka 8: Role

Název	Datový typ	Klíč	NULL	Index	Popis
postavaId	int(11)	ano	ne	ano	primární klíč
jmeno	char(50)	ne	ano	ne	jméno postavy
text	char(200)	ne	ano	ne	vlastnost postavy
zivoty	int(11)	ne	ano	ne	počet životů

Tabulka 9: Postava

Název	Datový typ	Klíč	NULL	Index	Popis
kartaId	int(11)	ano	ne	ano	primární klíč
jmeno	char(50)	ne	ano	ne	jméno karta
ceskeJmeno	char(50)	ne	ano	ne	české jméno karty
barva	char(10)	ne	ano	ne	barva rámečku
oznaceni	char(10)	ne	ano	ne	označené
text	char(200)	ne	ano	ne	text na kartě

Tabulka 10: HraciKarta

D Pravidla

Pravidla jsou citována z [3].

D.1 Postavy a cíl hry

Cíl hry každého hráče záleží na roli, kterou si na začátku hry vylosuje. Šerif musí zabít všechny, zločince a odpadlíka. Bandité musí zabít šerifa dříve, než šerif zabije je. Pomocníci pomáhají šerifovi a vyhrají, jen vyhraje-li on. Odpadlík se chce stát novým šerifem. Musí zůstat jako poslední hráč ve hře, to znamená nejprve zabít bandity a pomocníky šerifa a nakonec i šerifa.

D.2 Příprava

Ke hře budete potřebovat tolik karet rolí, kolik je hráčů, a to:

- pro 4 hráče: šerif, odpadlík, 2 bandité;
- pro 5 hráčů: šerif, odpadlík, 2 bandité, 1 pomocník;
- pro 6 hráčů: šerif, odpadlík, 3 bandité, 1 pomocník;
- pro 7 hráčů: šerif, odpadlík, 3 bandité, 2 pomocníci.

Zamíchejte uvedenou kombinaci karet a rozdejte po jedné každému hráči, aby nikdo neviděl, kdo jakou má. Šerif svou kartu otočí, ukáže ji ostatním a nechá ji před sebou ležet již otočenou. Ostatní se na svou kartu podívají, ale uchovají ji v tajnosti.

Zamíchejte 16 karet postav a nechte každého si jednu vylosovat. (Pro pokročilé hráče: každý hráč si vylosuje dvě postavy, z nichž si jednu vybere.) Každý hráč ukáže jméno své postavy a sdělí ostatním i její zvláštní schopnost. Poté si každý vezme jednu kartu postavy navíc a položí ji tak, aby byly vidět náboje na zadní straně karty. Částečně ji zakryje svou kartou postavy, aby bylo vidět tolik nábojů, kolik je znázorněno u obrázku jeho postavy. V průběhu hry bude množství takto nezakrytých nábojů znázorňovat počet životů postavy.

Šerif začíná hru s jedním nábojem navíc: má-li na kartě postavy uvedeny 3 náboje, bude mít na spodní kartě odkryté čtyři.

Zbylé karty rolí a postav nebudete pro hru potřebovat.

Všech 80 hracích karet zamíchejte a na začátku hry rozdejte každému hráči tolik karet, kolik má znázorněno nábojů na kartě své postavy (šerif má jeden život navíc, ale karet dostane také tolik, kolik má nábojů na kartě postavy).

Zbylé karty položte doprostřed stolu jako lízací balíček. Každý hráč si může vzít pomocnou kartu s přehledem symbolů.

Postavy

Každá z postav má nějakou unikátní zvláštní schopnost, která ovlivňuje strategii hráče v průběhu hry. O tuto schopnost nelze v průběhu hry nijak přijít ani ji jakkoli změnit. U každé schopnosti je uvedeno, kdy ji lze využít, ale hráč není povinen ji využít vždy, když k tomu má příležitost. Přehled schopností naleznete v kapitole Postavy na konci pravidel. Počet nezakrytých nábojů na spodní kartě znázorňuje počet životů (zranění) nutných k vyřazení postavy ze hry a také nejvyšší počet karet, které může hráč této postavy mít na konci svého tahu (viz třetí fáze kola).

D.3 Vlastní hra

Hru začíná šerif a hra pokračuje po směru hodinových ručiček. Tah každého hráče je rozdělen do následující 3 fází:

1. Líznutí dvou karet;
2. Zahrání libovolného počtu hracích karet;
3. Odhození přebývajících karet.

Líznutí dvou karet

Na začátku svého tahu si hráč z lízacího balíčku vezme dvě karty do ruky tak, aby je ostatní neviděli. Dojdou-li karty v balíčku, zamíchá se odhazovací balíček a použije se jako nový lízací balíček.

Zahrání libovolného počtu hracích karet

Hráč může zahrát karty ku svému prospěchu nebo proti jiným hráčům ve snaze vyřadit je ze hry. Hráč nemusí hrát žádné karty během svého tahu, stejně ale může zahrát tolik karet, kolik chce s následujícími omezeními: lze zahrát pouze jednu kartu BANG! za tah; nikdo před sebou nesmí mít vyložené dvě stejné modré karty.

Pokud chce zahrát kartu, řídí se symboly, které jsou vyobrazené na každé z nich. Tyto symboly jsou popsány dále. Karty můžete zahrát pouze během svého tahu (výjimkou jsou karty Pivo a Vedle!). Každá zahraná karta účinkuje okamžitě a je poté odhozena na odhazovací balíček. Výjimkou jsou karty s modrým okrajem, jako například zbraň na obrázku, které mají dlouhodobější efekt. Tyto modré karty zůstávají na stole lícem nahoru před hráčem, kterému patří.

Tyto karty platí dokud nejsou nějakým způsobem odstraněny ze hry (např. zahráním karty Cat Balou) nebo splněním speciální podmínky (např. Vězení nebo Dynamit).

Odhození přebývajících karet

Poté, co hráč již nechce, nebo nemůže hrát další karty, nastává třetí fáze jeho tahu. Hráč musí odhodit přebytečné karty tak, aby mu na konci tahu zbylo v ruce pouze tolik karet,

kolik má právě teď životů (nezakrytých nábojů na spodní kartě). Poté pokračuje další hráč ve směru hodinových ručiček.

Vyřazení postavy ze hry

Ztratí-li postava svůj poslední život, je vyřazena ze hry a hra pro jejího hráče končí, nezahraje-li ihned kartu Pivo (viz dále). Jakmile je hráč vyřazen ze hry, ukáže ostatním jakou hrál roli a odhodí všechny své hrací karty do odhazovacího balíčku.

Tresty a odměny

- Pokud šerif vyřadí svého pomocníka, musí odhodit všechny karty z ruky i před sebou.
- Ten, kdo vyřadí ze hry banditu (i kdyby ho vyřadil sám bandita), si jako odměnu lízne 3 karty z balíčku.

D.4 Konec hry

Hra končí, nastane-li jedna z následujících podmínek:

- Šerif je zabit. Odpadlík vyhrál, pokud je jediným kdo je ještě naživu. Jinak vyhrávají bandité.
- Všichni bandité a odpadlík jsou zabiti. Vítězem je šerif a jeho pomocníci.

D.5 Karty

Vzdálenost mezi hráči

Vzdálenost mezi dvěma hráči je minimum počtu míst mezi nimi, počítaje po směru nebo proti směru hodinových ručiček. Vzdálenost je dost důležitá, jelikož běžně hráč dostřelí pouze na hráče ve vzdálenosti 1. Je-li hráč vyřazen ze hry, nezapočítává se jeho místo do vzdálenosti mezi hráči. Takto se v průběhu hry vzdálenost mezi hráči zmenšuje.

Dvě karty upravují vzdálenost mezi hráči:

- Mustang: Hráč, který má před sebou vyloženou kartu Mustang, je viděn ostatními hráči na vzdálenost o jedna vyšší, tudíž je dál. On však na ostatní hráče vidí stále na normální vzdálenost.
- Appaloosa: Hráč, který má ve hře kartu Appaloosa, vidí ostatní hráče na vzdálenost o jedna menší. Ostatní hráči ho však stále vidí na normální vzdálenost. Vzdálenosti menší než 1 se počítají jako 1.

Zbraně

Každý hráč může na začátku hry střílet do hráče na vzdálenost 1, tedy do hráče, který sedí hned po jeho pravici nebo levici (je to tím, že každý hráč od začátku hry třímá Colt 45, ačkoli tato zbraň není znázorněná kartou).

Chtějí-li hráči střílet na větší vzdálenost, potřebují mít ve hře některou ze zbraní. Karty zbraní mají modrý okraj, černobílou ilustraci zbraně a číslo v hledáčku, které znázorňuje maximální dosažitelný dostřel. Zbraň ve hře nahrazuje Colt 45, dokud její karta není nějakým způsobem odstraněna (např. zahráním karty Panika! nebo Cat Balou). Hráč může mít ve hře pouze jednu kartu zbraně: chce-li zahrát jinou zbraň, musí tu dříve vyloženou odhodit. Nezapomeňte, že i když nemáte ve hře žádnou zbraň, stále třímáte Colt 45 a můžete střílet.

BANG! a Vedle!

Karty BANG! jsou základním prostředkem pro střelbu a snižování počtu životů ostatních hráčů. Chcete-li zahrát zahrát kartu BANG!, tedy vystřelit na jiného hráče, musíte si ověřit: a) jaká je vzdálenost mezi Vámi a Vaším cílem a b) jste-li schopni svou aktuální zbraní na tuto vzdálenost dostřelit.

Hráč zasažený výstřelem karty BANG! (ten, na kterého byla karta zahrána) může okamžitě zahrát kartu Vedle!, aby výstřelu uhnul. Nezahraje-li Vedle!, ztrácí jeden život (změnu musí zaznamenat posunutím karty své postavy, aby zakryl o jeden náboj více). Pokud jsou zakryty všechny náboje (tzn. ztrácí právě poslední život), hra pro něj končí, nezahraje-li okamžitě kartu Pivo (viz následující odstavec). Hráči mohou uhýbat pouze výstřelům namířeným proti nim. Karta BANG! je vždy odhozena, ačkoli výstřel minul cíl, takže střelec okamžitě po odehrání odhodí kartu Bang! a jeho protivník odhodí kartu Vedle!

Pivo

Tato karta vrátí hráči jeden život (posuňte kartu postavy, aby bylo vidět o náboj více). Hráči nemohou získat větší počet životů, než je jejich počáteční množství (všichni hráči podle počtu nábojů na kartě postavy, šerif o jeden život více)! Karta Pivo nemůže být použita k vyléčení jiného hráče.

Kartu Pivo lze zahrát dvěma způsoby: a) během vlastního tahu b) během cizího tahu, a to pouze v případě, když ztrácíte poslední život (tj. nestačí, když „jen“ ztrácíte jeden z životů).

Líznutí

Některé karty mají na sobě symbol karty a vedle něj znak, ukazující účinek této karty. Hráč, který má vyloženou takovou kartu před sebou, si musí nejdříve „líznout“, tj. otočit vrchní kartu balíčku a odhodit ji. Pokud znak na otočené kartě v levém dolním rohu bude odpovídat vyznačené barvě (popřípadě ještě hodnotě), bude platit původní karta. V

opačném případě se nic nestane: hráč neměl štěstí! Je-li na kartě uvedeno rozmezí hodnot, pak hráč musí otočit kartu s hodnotou zapadající do požadovaného rozmezí a barvy.

Speciální karty

Dynamit: Hráč, který chce zahrát tuto kartu, ji položí před sebe. Dynamit zůstane před ním ležet celé jedno kolo (doutná). Na začátku svého následujícího tahu si hráč před první fází musí „líznout" (otočit a odhodit vrchní kartu z balíčku). Otočí-li kartu mezi dvojkou až devítkou listů včetně, Dynamit exploduje, (odhodí se na vyhazovací balíček) a hráč ztrácí 3 životy. V opačném případě pošle Dynamit hráči po levici, který musí provést stejný test na začátku svého tahu. Hráči si takto předávají Dynamit dokola dokud někomu neexploduje nebo je odhozen pomocí karet Cat Balou nebo Panika!. Je-li Dynamit u hráče společně s kartou Vězení, probíhá test nejdříve na Dynamit poté na Vězení. Pokud by byl hráč eliminován kartou Dynamit, jeho vyřazení ze hry není považováno jako zásluha jiného hráče (viz pravidla Trestů a odměn a schopnosti některých postav).

Duel: Hráč zahráním této karty vyzývá kteréhokoli hráče na duel (nezáleží na vzdálenosti). Hráči (vyzvaný a vyzývající) nastřídačku střílí jeden po druhém – odhazují kartu BANG! Vyzvaný hráč začíná. Ten, který BANG! neodhodí (buď nechce, nebo nemůže), ztrácí jeden život a duel končí.

Hokynářství: Hráč otočí tolik vrchních karet z balíčku, kolik je hrajících (živých) hráčů. Počínaje hráčem, který zahrál tuto kartu, se postupuje po směru hodinových ručiček a každý hráč si vybere jednu z otočených karet a vezme si ji do ruky.

Indiáni!: Každý hráč, kromě hráče, který tuto kartu zahrál, může odhodit kartu BANG! (zastřelit svého indiána), nebo ztratit život. Nelze zahrát kartu Vedle!, ani se zkoušet schovat za Barel.

Vězení: Tuto kartu je možné položit před jakéhokoli hráče, který je tím uvězněn. Uvězněný hráč si musí „líznout" před začátkem svého kola: otočí-li srdce, utekl z vězení, karta Vězení je odhozena na vyhazovací balíček a on pokračuje normálně ve svém tahu. V opačném případě odhodí Vězení na odhazovací balíček a ztrácí první a druhou fázi svého tahu, tudíž pouze musí odhodit přebytečné karty podle aktuálního počtu svých životů. Po celou dobu pobytu ve vězení zůstává možným terčem karet BANG! a může stále zahrávat karty Vedle! a Pivo mimo vlastní kolo. Karta Vězení nemůže být zahrána na šerifa.

Volcanic: Hráč, který má tuto zbraň ve hře může zahrát libovolný počet karet BANG! během svého tahu. Tyto karty BANG! mohou být zahrány na jeden nebo různé cíle, ale pouze na vzdálenost 1 (tuto vzdálenost mohou ovlivnit karty Appaloosa a Mustang plus některé z postav).